CPP 程式設計題

題目名稱：大數計算機

題目說明：

請以 C++ 程式語言設計大數計算機，支援 Integer(整數)、Decimal(小數)運算，而且還須支援變數功能，亦即可以定義變數及該變數的型態。

除此之外，請遵循以下運算法則：

運算子優先權		
優先權	運算子	相同層級的運算順序
1	( )	
2	!	由左至右
3	^	由右至左
4	+- (正負號)	由右至左
5	* /	由左至右
6	+ -	由左至右

- 整數與整數運算其結果為整數。
- 整數與浮點數運算其結果為浮點數(順序相反亦然)。
- 除了支援任意數之間的加減乘除之外，還要有 Power(冪次)及正整數的 Factorial(階乘)。
- 輸出 Decimal 時，請直接輸出至小數點後 100 位，不必四捨五入或者無條件進位等動作，Decimal 的小數請以分數實作，亦即：
  1.0 / 3.0 * 3 = 1.00000000…，而非 0.999999999999...。
- 程式必須可以讓使用者將任意運算式的值賦予變數，而且可以重新賦值，例如：
  Set Integer A = 100 + 5! + Power(5, 2)
  Set Integer A = 3
  Set Decimal A = 1.0
  A = A + A
- 輸入格式自訂，測資只會給算式，Demo 的時候請自己輸入。
- 輸出格式自訂，以方便助教閱讀為原則。
- 輸出時機
  - Set Integer A = 3 // 不用輸出
  - A = 1 + 5 // 不用輸出
  - A // 輸出 6
  - A + A // 輸出 12
  - 1 * 33 + 4 // 輸出 37
- 當輸入非法運算式 (Ex: 除 0、小數的階乘)，需要輸出錯誤資訊，不能直接讓程式崩潰。

評分項目：

說明	範例	基本分數	額外分數（可帶入變數運算）
基本功能(20%)			
任意四則運算式(加減乘除以及括弧運算)，且滿足先乘除後加減，由左至右的求值順序。	Ex：1.5 + 3 * (-(-5))	無套用變數的情形下，完全正確才可繼續往下評分。否則 0 分。	
將精確度較高的型別指派給精確度較低型別之變數，自動捨去高精度資訊，例如將小數指派給整數，直接捨去所有小數點。 變數可以重新指派以及賦予任意運算式。 變數名稱可為任何長度的英文+數字，不會是關鍵字（Set , Integer , Decimal , …)，數字不會在字首。	Ex： Set Integer A=3.5 //這樣 A 會是 3	5%	
基本冪次運算: 任意運算式的冪次運算，Power(a, b) 代表 a^b，其中 a,b 都可以是任意運算式的結果，b 只有可能是 0.5 的整數倍，並且可以結合到一般運算中。冪次運算的結果不會有虛數。	Ex： ((2^3)^(0.5))^(2*2) 或 Power(Power(Power(2,3),0.5),2*2) 亦可。 Ex：1 + (2^3)	2.5%	5%
任意運算式的階乘運算 a!，其中 a 必為任意整數的運算結果(也必為正整數)，並且可以結合到一般運算中。	Ex：(2+4)! Ex：1 + (3 * 5) !	2.5%	5%
計算小數功能(40%)			
以分數格式儲存。 (所有基本功能測試完畢後確認無誤才給分)	Ex：1/3.0 * 3 答案為 1	15%	
運算輸出時精確度為 100 位。 (所有基本功能測試完畢後，確認無誤才給分)。	Ex：輸出 1/3.0 = 0.33333 … 3 // 有 100 個 3	15%	
可將小數運算結果套用到階乘以及冪次運算中。 其中階乘運算的部分，需要判斷小數運算的結果是否為整數。	Ex： ((2^3)^(1/2.0))^(2*0.5) 或者 Power(Power(Power(2,3),1/2.0),2*0.5)。 Ex：(1/3.0 * 6)!	10%	

API 設計(45%)		
為 Decimal、Integer 重載 operator <<, >>, +, -, *, /，使這兩種型別可以進行複合運算，並且可以使用 std::cout 以及 std::cin 進行輸出輸入。	Integer i; Decimal d; cin >> i >> d; cout << i + d;	20%
為 Decimal、Integer 重載 Constructor，可以直接賦予字串進行初始化，其中字串內容可以是任意運算式。	Integer i = "12345"; Decimal d = "0.3 * 3";	10%

繼承共同基底類別或者使用其它設計技巧，使得兩種型別可以放入同一容器中，並且可以走訪所有儲存的元素然後加以輸出。	Integer i = "123";
	Decimal d = "123.3";
	
	vector<NumberObject*> nums;
	nums.push_back(&i);
	nums.push_back(&d);
	for(const auto& num : nums)
	cout << *num << endl;
		15%

使用者體驗(0~25%)		
防呆測試，一個測試 2 %。		0~10%
呈現方式(美化 Console 或者以 C++開發 GUI)。		0~15%

備註：

- 部分基本功能細分為可帶入變數或者只能直接輸入 Constant 來計分。
- 1% 代表 1 分，超過 100 分以 100 分計算。